**Formale Methoden der Informatik WS 2010/2011**
**Lehrstuhl für Datenbanken und Künstliche Intelligenz**
Prof. Dr. Dr. F. J. Radermacher • H. Ünver • T. Rehfeld • J. Dollinger

**8. Aufgabenblatt**
Besprechung in den Tutorien vom 22.12.2010 (ab Übungstermin)
bis 12.01.2011 (bis Übungstermin)

## 39. Aufgabe (5 Punkte) Verständnisfragen:

a) Was ist eine Iteration? ($\frac{1}{2}$ Punkt)

in der (numerischen) Mathematik:

Wiederholtes Anwenden desselben mathematischen Verfahrens, um ein Problem schrittweise, aber zielgerichtet anzunähern.

in der Informatik:

Mehrmaliges Ausführen einer Aktion. Häufig zählt man dabei mit, zum wievielten Mal die Aktion ausgeführt wird (muss aber nicht sein).
Enstprechung im Pseudocode: WIEDERHOLE SOLANGE ...

b) Was ist eine Rekursion? ($\frac{1}{2}$ Punkt)

in der Mathematik:

*meist:* Eine Definition, die sich auf sich selbst bezieht.

in der Informatik:

Eine Verfahrensvorschrift (Algorithmus, Funktion, etc.), die sich (entweder direkt oder indirekt) selbst aufruft.
Enstprechung im Pseudocode: Aufruf der Funktion selbst, die gerade geschrieben wird.

c) Aus welchen Teilen besteht eine Rekursion? ($\frac{1}{2}$ Punkt)

Rekursionsanfang, Rekursionsschritt – oder äquivalente Antworten.
(z.B.: Abbruchbedingung (= Basisfall), eigentliche Rekursion (= Selbstaufruf))

d) Was ist der Unterschied zwischen Rekursion und Iteration? ($\frac{1}{2}$ Punkt)

Gemeinsamkeiten:

In beiden Fällen werden Aktionen mehrmals ausgeführt,

Unterschiede:

in einem Fall (Iteration) durch Schleifen (explizite Wiederholungsanweisungen),

```
Pizzaessen:
  Wiederhole vom ersten bis zum letzen Bissen:
    schneide diesen Bissen ab und iss ihn auf
  // Ende von Wiederhole
// Ende von Pizzaessen
```

im anderen Fall (Rekursion) durch Selbstaufrufe (oder Selbstbezüge).

```
Pizzaessen:
  wenn Teller leer: FERTIG
  sonst:
    schneide einen Bissen ab und iss ihn auf
    Pizzaessen
  // Ende von wenn
// Ende von Pizzaessen
```

e) Wie funktioniert der Beweis durch Induktion? ($\frac{1}{2}$ Punkt)

Idee:

Ist bekannt,

- dass eine bestimmte von $n$ abhngige Aussage für $n = 1$ gilt (Induktionsanfang) und
- dass für jede beliebige natürliche Zahl $k$ aus der Gültigkeit der Aussage für $n = k$ (Induktionsannahme oder Induktionsvoraussetzung) auch die Gültigkeit für $n = k + 1$ folgt (Induktionsschritt oder Induktionsschluss),

dann folgt (nach dem Induktionsaktion – also der Annahme, dass dies alles auch so funktioniert), dass diese Aussage für *alle* natürlichen Zahlen $n$ gilt.

f) Welche Gemeinsamkeiten weisen Rekursion und der Beweis durch Induktion zueinander auf? ($\frac{1}{2}$ Punkt)

Ähnlich wie bei der Rekursion wird beim Beweis durch vollständige Induktion von einem Basisfall ausgegangen (Induktionsanfang).

Für alle anderen Fälle wird die (bisher nur angenommene) Gültigkeit für den Fall $n$ auf den Nachfolgefall $n + 1$ erweitert.

Genauer: Unter der Annahme, dass die zu zeigende Behauptung für $n$ stimmt, muss der Schritt von $n$ auf $n + 1$ nachgewiesen werden (Induktionsschritt).

Dann, so argumentiert man, lässt sich vom Basisfall ($n = 1$) auf seinen direkten Nachfolger ($n = 2$) schliessen und dann wieder auf dessen Nachfolger ($n = 3$), etc. – so dass schliesslich der gesamte Zahlenraum (der Natürlichen Zahlen) abgedeckt ist.

Die Gemeinsamkeit zwischen Rekursion und vollständiger Induktion ist letztendlich die Selbstbezüglichkeit. Die Rekursion führt ihre eigene Definition auf sich selbst zurück, indem sie mit jedem Schritt das (implizit) „Problem“ ein wenig „verkleinert“, bis schließlich ein sogenannter Basisfall (auch genannt Abbruchbedingung) erreicht ist.

Die vollständige Induktion greift auf die vorher aufgestellte Behauptung (Induktionsbehauptung) zurück und zeigt mit jedem Schritt (implizit), dass sich die Behauptung auf den gesamten Zahlenraum anwenden lässt.

g) Was ist eine Abbruchsbedingung? ($\frac{1}{2}$ Punkt)

Jede rekursive Funktion benötigt (neben dem eigentlichen rekursiven Selbst-Aufruf) eine Abbruchbedingung (auch genannt: Basisfall). Andernfalls würde sie niemals aufhören, sich immer wieder selbst aufzurufen und käme niemals zu einem Ergebnis.

Die Abbruchbedingung ist die Bedingung, mit der der wiederholte Selbst-Aufruf einer Rekursion abbricht. Meist wird hier ein zurückgeliefertes Ergebnis einfach „per Definition“ gesetzt – oder schlicht auf andere als auf rekursive Art ermittelt.

h) Was ist ein (Band-)Alphabet? ($\frac{1}{2}$ Punkt)

Eine Menge zulässiger Symbole (hier: Zeichen), die (z.B. auf dem Band einer Turingmaschine) vorkommen können.

i) Was ist eine Turingmaschine? (1 Punkt)

Ein Maschinen**modell** zur Untersuchung von Berechenbarkeit.

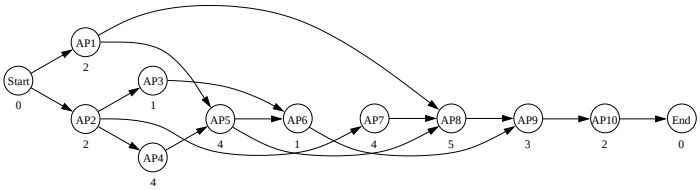
Zustandsmaschine (Alphabet von möglichen Zuständen) mit unendlichem Band und Lese-/Schreibkopf zum “Speichern” von Informationen (Alphabet von möglichen Belegungen / konkrete Vorbelegung), sowie Übergangsfunktionen zum Wechsel der Zustände.

## 40. Aufgabe (6 Punkte) Netzplan und Kritischer Pfad:

Die Entwicklung einer neuen Generation von Mobiltelefonen wird durch unterschiedliche Arbeitspakete realisiert. Den Arbeitspaketen werden Aufgaben zugeordnet, die die erfolgreiche Entwicklung erleichtern sollen. Die nachfolgende Tabelle zeigt die hierfür nötigen Arbeitspakete mit der jeweiligen Dauer pro Arbeitspaket. Dabei werden die jeweils unmittelbaren Vorgänger eines Arbeitspakets in der Vorgangsliste dokumentiert.

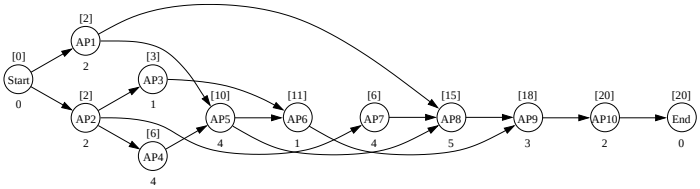
| **Vorgang** | | **Dauer [Monate]** | **vorangehende Vorgänge** |
|---|---|---|---|
| AP 1: | Prototyp Handy | 2 | – |
| AP 2: | Anforderungen Software | 2 | – |
| AP 3: | Erstellung Testplan | 1 | AP 2 |
| AP 4: | Entwurf SW | 4 | AP 2 |
| AP 5: | Implementierung SW | 4 | AP 1, AP 4 |
| AP 6: | Test SW | 1 | AP 3, AP 5 |
| AP 7: | Entwurf Handy HW | 4 | AP 2 |
| AP 8: | Bau Handy HW | 5 | AP 1, AP 5, AP 7 |
| AP 9: | Integration | 3 | AP 6, AP 8 |
| AP 10: | Systemtest | 2 | AP 9 |

a) Erstellen Sie einen Graphen (Netzplan), der den Projektplan darstellt und der die Abhängigkeiten der einzelnen Arbeitspakete untereinander berücksichtigt.



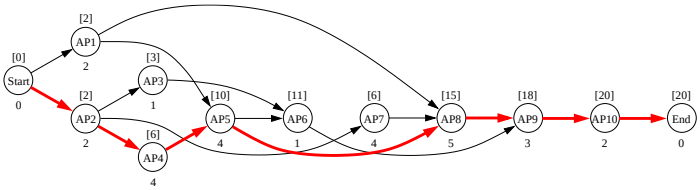

Die Knoten „Start“ und „End“ sind reine Hilfsknoten (Dummy-Knoten) und benötigen keinen eigenen Zeitaufwand. Den Knoten „End“ könnte man im Prinzip auch weglassen.

b) Ermitteln Sie den Zeitaufwand der einzelnen Pfade.




Zum Graphen aus Aufgabenteil a) kommen nun noch Zeitangaben (in eckigen Klammern) hinzu. Die Zeitangabe eines Knotens wird gebildet aus dem *Maximum* der Zeitangaben aller seiner Vorgänger, zu welchem der eigene Zeitaufwand hinzugezählt wird.

c) Ermitteln Sie den kritischen Pfad.




Den Kritischen Pfad erhält man, indem man vom Zielknoten ausgehend zurückläuft und unter den Vorgängerkanten jeweils die Kante mit dem bisher höchsten Zeitaufwand auswählt (bzw. die Kante, die zum Knoten mit dem bisherigen höchsten Zeitaufwand führt). Es kann mehr als einen Kritischen Pfad geben, wenn alle Kritischen Pfade den gleichen Zeitaufwand benötigen.

## 41. Aufgabe (6 Punkte) Fibonacci-Zahlen:

In der Vorlesung haben Sie die Fibonacci-Zahlen behandelt.

a) Eine einfache *(nicht rekursive)* Methode, die Fibonacci-Folge zu berechnen, ist folgende:

Man erstelle eine Tabelle mit den Fibonacci-Zahlen. Die ersten beiden Zeilen sind gegeben. Alle weiteren errechnen sich durch Addition der letzten beiden Werte.

$$\begin{array}{rcl} fib(0) & = & 0 \\ fib(1) & = & 1 \\ fib(2) & = & \\ fib(3) & = & \\ fib(4) & = & \\ & \vdots & \end{array}$$

Berechnen Sie auf diese Weise *(noch nicht rekursiv)* die Fibonacci-Zahlen von $fib(2)$ bis $fib(10)$.

$$\begin{array}{rcl} fib(0) & = & 0 \\ fib(1) & = & 1 \\ fib(2) & = & 1 \\ fib(3) & = & 2 \\ fib(4) & = & 3 \\ fib(5) & = & 5 \\ fib(6) & = & 8 \\ fib(7) & = & 13 \\ fib(8) & = & 21 \\ fib(9) & = & 34 \\ fib(10) & = & 55 \end{array}$$

b) Mit Hilfe der in der Vorlesung angegebenen geschlossenen Formel

$$fib'(n) \quad = \quad \frac{1}{\sqrt{5}} \cdot \left[ \left( \frac{1+\sqrt{5}}{2} \right)^n - \left( \frac{1-\sqrt{5}}{2} \right)^n \right]$$

berechnen Sie: $fib'(4)$

Hinweis: Bitte Lösungsweg mit angeben. „Taschenrechner-Lösungen“ sind nicht erwünscht.

es gilt:
$$\begin{array}{rcl} (a+b)^4 & = & a^4 + 4a^3b + 6a^2b^2 + 4ab^3 + b^4 \\ (a-b)^4 & = & a^4 - 4a^3b + 6a^2b^2 - 4ab^3 + b^4 \end{array}$$

$$\begin{array}{rcl} fib'(4) & = & \frac{1}{\sqrt{5}} \cdot \left[ \left( \frac{1+\sqrt{5}}{2} \right)^4 - \left( \frac{1-\sqrt{5}}{2} \right)^4 \right] \\ & = & \frac{1}{\sqrt{5}} \cdot \left[ \frac{\left(1+\sqrt{5}\right)^4}{2^4} - \frac{\left(1-\sqrt{5}\right)^4}{2^4} \right] \\ & = & \frac{1}{\sqrt{5}} \cdot \left[ \frac{1^4 + 4\sqrt{5} + 6\sqrt{5}^2 + 4\sqrt{5}^3 + \sqrt{5}^4}{16} - \frac{1^4 - 4\sqrt{5} + 6\sqrt{5}^2 - 4\sqrt{5}^3 + \sqrt{5}^4}{16} \right] \\ & = & \frac{1}{\sqrt{5}} \cdot \left[ 2 \cdot \frac{4\sqrt{5} + 4\sqrt{5}^3}{16} \right] \\ & = & \frac{1}{\sqrt{5}} \cdot \left[ \frac{4\sqrt{5} + 4 \cdot 5\sqrt{5}}{8} \right] \\ & = & \frac{4+20}{8} \quad = \quad \frac{24}{8} \quad = \quad 3 \end{array}$$

c) Mit Hilfe der in der Vorlesung angegebenen Rekursionsformel

$$fib''(n) \quad = \quad \begin{cases} 0 & , \text{ falls } n = 0 \\ 1 & , \text{ falls } n = 1 \\ fib''(n-1) + fib''(n-2) & , \text{ falls } n > 1 \end{cases}$$

berechnen Sie: $fib''(5)$

$$\begin{aligned}
fib''(5) &= fib''(4) + fib''(3) \\
&= fib''(3) + fib''(2) \quad + \quad fib''(2) + \underbrace{fib''(1)}_{1} \\
&= fib''(2) + \underbrace{fib''(1)}_{1} \quad + \quad \underbrace{fib''(1)}_{1} + \underbrace{fib''(0)}_{0} \quad + \quad \underbrace{fib''(1)}_{1} + \underbrace{fib''(0)}_{0} \quad + \quad 1 \\
&= \underbrace{fib''(1)}_{1} + \underbrace{fib''(0)}_{0} + 1 \quad + \quad 1 + 0 \quad + \quad 1 + 0 \quad + \quad 1 \\
&= 1 + 0 + 1 \quad + \quad 1 + 0 \quad + \quad 1 + 0 \quad + \quad 1 \\
&= 5
\end{aligned}$$

## 42. Aufgabe (10 Punkte) Handsimulationen:

Gegeben seien die folgenden Funktion:

a) $mod(a,b) = \begin{cases} a & , \text{ falls } a < b \\ mod(a-b,b) & , \text{ sonst} \end{cases}$

Berechnen Sie:

(a) $mod(18,6) = mod(12,6) = mod(6,6) = mod(0,6) = 0$

(b) $mod(18,42) = 18$

(c) $mod(20,4) = mod(16,4) = mod(12,4) = mod(8,4)$
$= mod(4,4) = mod(0,4) = 0$

(d) $mod(24,20) = mod(4,20) = 4$

(e) $mod(24,68) = 24$

(f) $mod(42,18) = mod(24,18) = mod(6,18) = 6$

(g) $mod(68,24) = mod(44,24) = mod(20,24) = 20$

b) $ggT(a,b) = \begin{cases} b & , \text{ falls } mod(a,b) = 0 \\ ggT(b,mod(a,b)) & , \text{ sonst} \end{cases}$

Berechnen Sie:

(a) $ggT(18,42) = ggT(42,\underbrace{mod(18,42)}_{18})$ (siehe 38.a.ii)
$= ggT(18,\underbrace{mod(42,18)}_{6})$ (siehe 38.a.vi)
$= 6$ (siehe 38.a.i) $mod(18,6) = 0$

(b) $ggT(24,68) = ggT(68,\underbrace{mod(24,68)}_{24})$ (siehe 38.a.vi)
$= ggT(24,\underbrace{mod(68,24)}_{20})$ (siehe 38.a.vii)
$= ggT(20,\underbrace{mod(24,20)}_{4})$ (siehe 38.a.iv)
$= 4$ (siehe 38.a.iii) $mod(20,4) = 0$

(c) $ggT(42,18) = 6$ (siehe 38.b.i, erstes Zwischenergebnis)

alternativ:

$ggT(42,18) = ggT(6,\underbrace{mod(42,18)}_{6})$ (siehe 38.b.vi)
$= ggT(6,\underbrace{mod(6,6)}_{0})$
$= 6$

## 43. Aufgabe (4 Punkte)

Gegeben Sei der Pseudocode für die Fakultät:

$$n! \equiv fact(n) := \begin{cases} 1 & \text{, falls } n = 1 \\ n \cdot fact(n-1) & \text{, sonst} \end{cases}$$

Führen Sie eine Handsimulation anhand des nachfolgenden Pseudocodes durch und berechnen Sie in beiden Fällen $fact(4)$.

a) Rekursiv:

```
fact(n):
    wenn n = 1:
        Ergebnis ist: 1                    // Abbruchbedingung oder Basisfall
    sonst
        Ergebnis ist: n · fact(n − 1)      // Selbstaufruf – Rekursion
    // Ende von wenn
// Ende von fact(n)
```

$fact(4): \quad (\rightarrow n = 4)$
wenn $n = 1:$ (falsch)
sonst
Ergebnis ist: $n \cdot fact(n-1)$
$= \quad 4 \cdot fact(3)$

NR: $fact(3): \quad (\rightarrow n = 3)$
wenn $n = 1:$ (falsch)
sonst
Ergebnis ist: $n \cdot fact(n-1)$
$= \quad 3 \cdot fact(2)$

NR: $fact(2): \quad (\rightarrow n = 2)$
wenn $n = 1:$ (falsch)
sonst
Ergebnis ist: $n \cdot fact(n-1)$
$= \quad 2 \cdot fact(1)$

NR: $fact(1): \quad (\rightarrow n = 1)$
wenn $n = 1:$ (wahr)
Ergebnis ist: 1

$= \quad 2 \cdot 1$
$= \quad 2$ (Zwischenergebnis)

$= \quad 3 \cdot 2$
$= \quad 6$ (Zwischenergebnis)

$= \quad 4 \cdot 6$
$= \quad 24$ (Gesamtergebnis)

b) Iterativ:

$fact(n)$:

*produkt* $:= n$

wiederhole solange $n > 1$ :
  $n := n - 1$
  *produkt* $:=$ *produkt* $\cdot\, n$
// Ende von wiederhole

} Schleife
} Wiederholung innerhalb der Funktion

Ergebnis ist: *produkt*

// Ende von $fact(n)$

$fact(4)$:

$n = 4$
*produkt* $:= n = 4$
wiederhole solange $\underbrace{n}_{4} > 1$: (wahr)

$n := n - 1 = 3$
*produkt* $:= \underbrace{\textit{produkt}}_{4} \cdot \underbrace{n}_{3} = 4 \cdot 3 = 12$

nächster Durchlauf: $\underbrace{n}_{3} > 1$ (wahr)

$n := n - 1 = 2$
*produkt* $:= \underbrace{\textit{produkt}}_{12} \cdot \underbrace{n}_{2} = 12 \cdot 2 = 24$

nächster Durchlauf: $\underbrace{n}_{2} > 1$ (wahr)

$n := n - 1 = 1$
*produkt* $:= \underbrace{\textit{produkt}}_{24} \cdot \underbrace{n}_{1} = 24 \cdot 1 = 24$

kein nächster Durchlauf, da $n > 1$ (falsch)

Ergebnis ist: $\underbrace{\textit{produkt} = 24}_{\text{Gesamtergebnis}}$

## 44. Aufgabe (6 Punkte)

a) Gegeben sei die folgende Funktion:

$$f(n,m) = \begin{cases} 0 & , \text{ falls } m = 0 \\ f(n,m-1)+n & , \text{ sonst} \end{cases}$$

Zeigen Sie durch Induktion oder widerlegen Sie: $f(n,m) = n \cdot m$
(Multiplikation zweier Zahlen $n$ und $m$)

| | | | | | |
|---|---|---|---|---|---|
| Ind. Anfang: | | $f(n,0)$ | $=$ | $0$ | |
| Ind. Beh.: | | $f(n,m)$ | $=$ | $n \cdot m$ | |
| Ind. Schritt: | es gilt: | $f(n,m-1)$ | $=$ | $n \cdot (m-1)$ | |
| $(m-1 \to m)$ | zu zeigen: | $f(n,m)$ | $=$ | $f(n,m-1)+n$ | |
| | | $f(n,m)$ | $=$ | $n \cdot m$ | |
| | $\Rightarrow$ | $f(n,m-1)$ | $=$ | $n \cdot (m-1)$ | |
| | | | $=$ | $n \cdot m - n$ | $\mid +n$ |
| | | $f(n,m-1)+n$ | $=$ | $n \cdot m$ | q.e.d. |

b) Stellen Sie eine Rekursionsformel auf für die Addition zweier Zahlen $n$ und $m$.

$$f(n,m) \;=\; \begin{cases} n & , \text{ falls } m = 0 \\ f(n,m-1)+1 & , \text{ sonst} \end{cases}$$

oder auch

$$\begin{aligned} f(n,0) &= n \\ f(n,m) &= f(n,m-1)+1 \end{aligned}$$

c) Stellen Sie eine Rekursionsformel auf für das Minimum zweier Zahlen $n$ und $m$.

$$min(n,m) \;=\; \begin{cases} 0 & , \text{ falls } m = 0 \text{ oder } n = 0 \\ min(n-1,m-1)+1 & , \text{ sonst} \end{cases}$$

oder auch

$$\begin{aligned} min(0,m) &= 0 \\ min(n,0) &= 0 \\ min(n,m) &= min(n-1,m-1)+1 \end{aligned}$$

## 45. Aufgabe (3 Punkte) Terminierung

Terminiert (hält) das folgende Programm für alle möglichen Eingaben?

Eingabe $x$
Wiederhole solange gilt: $x$ ist nicht teilbar durch 3
    $x := x + 7$
*// Ende der Wiederholung*
Ausgabe $x$

Hinweis:

- $x \in \mathbb{N}$
- **Teilbar** bedeutet: es muss ein **ganzzahliges** Ergebnis bei der Division entstehen.

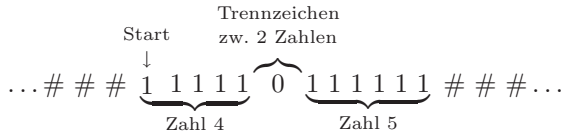
Ja!
Man kann mehrere Fälle unterscheiden:

1. $x$ **ist durch 3 teilbar:** $(mod(x, 3) = 0)$
   → dann terminiert das Programm sofort
   (d.h. es betritt nicht einmal die Wiederholungs-Schleife)
2. $x$ **ist um eins größer als eine durch drei teilbare Zahl:** $(mod(x, 3) = 1)$
   Die Zahl 7 selbst ist um eins größer als eine durch drei teilbare Zahl (nämlich 6).
   → dann ist `x + 7` um zwei größer als eine durch drei teilbare Zahl.
   (weiter mit Fall 3.)
3. $x$ **ist um zwei größer als eine durch drei teilbare Zahl:** $(mod(x, 3) = 2)$
   Die Zahl 7 selbst ist um eins größer als eine durch drei teilbare Zahl (nämlich immer noch 6).
   → dann ist $x + 7$ um drei größer als eine durch drei teilbare Zahl. Dies ist selbst aber wieder eine durch drei teilbare Zahl.

→ Nach spätestens zwei Durchläufen durch die Schleife ist die Schleifenbedingung nicht mehr erfüllt und das Programm wird nach der Schleife fortgesetzt.

## 46. Aufgabe (6 Punkte) Turingmaschine

Gegeben ist eine Turingmaschine, die eine Rechenoperation auf zwei natürlichen Zahlen ausführt, die, wie folgt, auf dem Eingabeband liegen:




Die Zahl 4 wird dabei als fünf 1-en dargestellt, damit auch noch die Zahl 0 als eine 1 dargestellt werden kann.

Wesentlich zur Beschreibung der Turingmaschine ist eine Überführungsfunktion, welche beschreibt, welche Aktionen sie ausführt, wenn der Lese-/Schreibkopf ein Zeichen vom Band liest und sich die Turingmaschine dabei in einem bestimmten inneren Zustand befindet.

| | 0 | 1 | # | *Eingabe* |
|---|---|---|---|---|
| $z_0$ | $1, L, z_1$ | $1, R, z_0$ | $\#, H, e$ | |
| $z_1$ | | $1, L, z_1$ | $\#, R, z_2$ | |
| $z_2$ | | $\#, R, z_3$ | | |
| $z_3$ | | $\#, R, z_4$ | | |
| $z_4$ | | $1, H, e$ | | |
| Zustände | | | | |

Dabei bedeutet: L der Lese-/Schreibkopf bewegt sich einen Schritt nach links,
R der Lese-/Schreibkopf bewegt sich einen Schritt nach rechts,
H die Turingmaschine geht in den Haltezustand über.

Lesart der Tabelle:

Wenn der Lese-/Schreibkopf eine 0 auf dem Band liest und sich die Turingmaschine dabei gerade im Zustand $z_0$ befindet,

- dann schreibe an diese Stelle (auf dem Band) eine 1,
- gehe einen Schritt nach links
- und wechsle in den inneren Zustand $z_1$

Die Turingmaschine befindet sich am Anfang im Zustand $z_0$ und der Lese-/Schreibkopf befindet sich zu Beginn am Anfang der ersten Zahl auf dem Band.

a) Machen Sie eine Handsimulation und vollziehen Sie Schritt für Schritt nach, was die Turingmaschine mit dem Ein-/Ausgabeband tut und in welche inneren Zustände sie dabei übergeht.

Beispielsweise:

$$\#\overset{z_0}{1}11110111111\# \vdash \#1\overset{z_0}{1}1110111111\# \vdash \#11\overset{z_0}{1}110111111\# \vdash \#111\overset{z_0}{1}10111111\# \vdash \#1111\overset{z_0}{1}0111111\# \vdash \#11111\overset{z_0}{0}111111\# \vdash \#1111\overset{z_1}{1}1111111\# \vdash \ldots$$

Wie geht es weiter?

$$\ldots \vdash \#111\overset{z_1}{1}11111111\# \vdash \#11\overset{z_1}{1}111111111\# \vdash \#1\overset{z_1}{1}1111111111\# \vdash \#\overset{z_1}{1}11111111111\# \vdash \#\overset{z_2}{1}11111111111\# \vdash \#\#\overset{z_3}{1}1111111111\# \vdash \#\#\#\overset{z_4}{1}111111111\# \vdash \#\#\#\overset{e}{1}111111111\#$$

b) Terminiert das Programm?

Offensichtlich, ja.

c) Interpretieren Sie das Ergebnis. Was tut das Programm?

Beobachtung:

Im Verlauf des Programmes wird zunächst die 0 zwischen den beiden Zahlen mit einer 1 überschrieben und dann auf der linken Seite zwei 1-en mit # (Blanks) überschrieben. Ergebnis ist eine Ziffernfolge von zehn 1-en.

Interpretation:

Nach der oben angegebenen Zahlendarstellung sind die Ursprünglichen beiden Zahlen auf dem Band zu interpretieren als 4 und 5 (jeweils eins weniger als 1-en hintereinander).

Das Endergebnis ist zu interpretieren als 9.

Es liegt nahe zu vermuten, dass die durchgeführte Rechenoperation die Addition der beiden Ausgangszahlen ist.